Semantic Modeling of Outdoor Scenes for the Creation of Virtual Environments and Simulations

Meida Chen
USC – Sonny Astani Department of Civil and Environmental Engineering
meidache@usc.edu

Ryan McAlinden, Ryan Spicer
USC – Institute for Creative Technologies
{mcalinden, spicer}@ict.usc.edu

Lucio Soibelman
USC – Sonny Astani Department of Civil and Environmental Engineering
soibelman@usc.edu

Abstract

Efforts from both academia and industry have adopted photogrammetric techniques to generate visually compelling 3D models for the creation of virtual environments and simulations. However, such generated meshes do not contain semantic information for distinguishing between objects. To allow both user- and system-level interaction with the meshes, and enhance the visual acuity of the scene, classifying the generated point clouds and associated meshes is a necessary step. This paper presents a point cloud/mesh classification and segmentation framework. The proposed framework provides a novel way of extracting object information – i.e., individual tree locations and related features while considering the data quality issues presented in a photogrammetric-generated point cloud. A case study has been conducted using data that were collected at the University of Southern California to evaluate the proposed framework.

1. Introduction

Over the past few years, recent advances in sensing technologies and computer vision algorithms, photogrammetric techniques have been highly studied. A 3D point cloud/mesh of an object generated by photogrammetry consists of detailed information on shapes and surface textures. Many existing studies and applications from both academia and industry have adopted photogrammetry to create as-is 3D models of outdoor scenes for different purposes such as urban planning, building energy simulation, virtual environments, historical building information storage, construction quality and schedule control, facility management, and so forth [1], [2]. With the rapid advancement of unmanned aerial vehicle (UAV) technology, the data collection process for creating 3D point clouds/meshes of an outdoor scene using photogrammetric techniques that can be conducted with few resources (people and equipment) in a short period of time has become feasible. The authors of this paper previously developed a UAV path-planning tool in which imagery data can be collected within two hours to model a 1km^2 area, and the 3D meshes can be reconstructed within a few hours [3]. Such a rapid 3D modeling process for an area of interest has been brought to the U.S. Army's attention and motivated the One World Terrain (OWT) Project. One of the objectives of the OWT is to provide small units with the organic capability to create geo-specific virtual environments for training and rehearsal purposes to support military operations. For more information about the OWT project, readers can refer to http://www.dronemapping.org/. The work presented in this paper is part of the OWT project.

A visually realistic 3D mesh can be generated through photogrammetric techniques to create virtual environments for immersive tools and technologies. However, the generated meshes simply contain the polygons and textures—i.e., they do not contain semantic information for distinguishing between objects such as the ground, buildings, and trees. Being able to segment, classify, and recognize distinct types of objects together, along with identifying and extracting associated features (e.g., individual tree locations) in the generated meshes, are essential tasks in creating realistic virtual simulations. Rendering different objects in a virtual environment and assigning actual physical properties to each will not only enhance the visual quality, but also allow various user interactions with a terrain model. For instance, consider the case of providing a user with the shortest path from location A to location B in which the individual is visible from a given vantage point. With an artificial intelligence (AI) searching algorithm, such as A*, the shortest path could be computed, and penalties cloud be assigned to a route based on the number of obstructions blocking the enemies' line-of-

URI: https://hdl.handle.net/10125/59634
ISBN: 978-0-9981331-2-6

sight. However, in reality, line-of-sight that is blocked by concrete walls, glass windows, and trees should be assigned different penalties when considering a route, since some materials cannot protect soldiers from gunshots (e.g., glass windows). Though the example is an oversimplification, it emphasizes the point that, without semantic segmentation of the mesh data, realistic virtual simulations cannot be achieved.

In this paper, a photogrammetric generated 3D point clouds/meshes segmentation and information extraction framework is proposed. The proposed framework utilizes both supervised and unsupervised machine learning algorithms. The segmentation process is first performed on the point clouds. Following that, since photogrammetric generated meshes are in the same coordinate system as the point cloud, the meshes are segmented according to the point cloud segmentation results. The rest of the paper is organized as follows. Section 2 provides the related literature and the identified research gaps. Section 3 discusses the research objective and questions. Section 4 introduces the proposed framework and the individual processes in details. Section 5 and 6 are the case study and conclusion respectively.

2. Background and Related Literature

In this section, the basic concepts of photogrammetry are introduced. Following that, previous studies that have focused on semantic labeling of point clouds and individual tree location identification are reviewed, and research gaps are highlighted.

2.1. Photogrammetry

Photogrammetry is an image-based point cloud/mesh creation technique, which is a reverse engineering process. Since 2D images do not have the depth information of a scene, this process recovers the depth information from pairs of images. Key points, which are known as features (e.g., Scale Invariant Feature Transforms [SIFT] or Speeded Up Robust Features [SURF]) are detected from each image. Following that, the same key points/features in different images are matched, upon which the camera orientation is then estimated. Finally, a dense point cloud is constructed through a triangulation process [6] that consists of millions of points representing the spatial information of object surfaces. The cloud also contains additional per point information—i.e., color. In order to use the data in modern game engines for simulation, 3D meshes are needed instead of point clouds. By connecting points in a point cloud to form triangular surfaces, a 3D triangular mesh can be generated [3]. Han and Golparvar-Fard used unordered site photos to reconstruct a point cloud for construction progress monitoring [7]. Several other studies have also used the photogrammetric technique to capture the as-is condition of outdoor scenes [8]-[12].

2.2. Point cloud segmentation/classification

3D point cloud segmentation, classification, and object recognition is the foundation of many cutting-edge technologies used in autonomous vehicles, forest structure assessment, and scan-to-BIM process, among others [13], [14]. Nevertheless, segmenting a large 3D point cloud with millions of point data into different categories is still a challenging task. Earlier works by A. Frome et al. proposed to segment a point cloud by using 3D shape descriptors for matching individual points with a 3D point cloud database sampled from 3D object models [15]. Many studies have made valuable contributions on segmenting LIDAR-collected point clouds. Researchers have proposed to combine local point descriptors (e.g., curvature, planarity, and density ratio, etc.) with LIDAR features (e.g., echo-based and waveform-based features) and used supervised machine-learning approaches to segment the point cloud into different classes [16]. Other researchers have proposed bare-earth extraction as a pre-process for segmenting other objects such as human-made structures, buildings, and cars [17]. A few studies have adopted deep learning approaches for point cloud feature attribution/segmentation in which inspiration was drawn from recent successes of image classification. Such approaches generally fall into two categories: (1) projecting a 3D point cloud onto a 2D plane and performing an image segmentation process [18]; and (2) performing the segmentation process directly on the 3D point cloud through deep learning [19]. However, linking the work of point cloud segmentation (especially photogrammetric-generated point clouds) and extracting detailed information, such as individual tree location for generating synthetic training environments, and allowing artificial intelligence path planning remains a challenge.

Many existing point cloud segmentation and classification approaches have been designed and tested with LIDAR point clouds. Segmenting photogrammetric-generated point clouds is much more challenging than segmenting LIDAR data due to the following two reasons. First, several point features that are available in the LIDAR data do not exist in the photogrammetric-generated point clouds (e.g., echo-based and waveform-based features). Second, the photogrammetric-generated point clouds tend to be

noisy, and in some cases, ground cannot be captured due to dense canopy [14]. Thus, it is crucial to evaluate the performance of existing approaches for photogrammetric-generated point cloud segmentation.

2.3. Individual tree location identification

Several studies have focused on individual tree segmentation in order to identify their locations and other related features from LIDAR-collected point clouds [20], [21]. Most proposed approaches contain two steps: (1) segmenting tree points from everything else; and (2) identifying individual tree locations and related features from the segmented tree points [22]-[24]. Huang et al. [22] and Zhang et al. [20] have proposed to segment tree points by combining the use of an airborne LIDAR-generated point cloud and near-infrared images. Vegetation regions were extracted using the normalized difference vegetation index (NDVI) derived from near-infrared images, and the extracted regions were then projected onto the point cloud to segment the tree points. Individual tree locations were extracted in a region-growing fashion with setting treetops (points that have local maximum height value) as seed points. Persson et al. [23] focused on identifying individual tree locations in dense forest areas. Trees were first segmented from the ground by using an active contours algorithm. Following that, individual tree locations were identified by fitting a parabolic surface to the top of the segmented tree canopies Ritter et al. [24] also focused on forest dataset. The authors proposed a two-step clustering algorithm, which exploited the ability of terrestrial laser scanners to collect data points on the leaves inside of crowns. In the first step, tree points were stratified into horizontal layers, and cluster centers in each horizontal layer were computed based on the point density. These centers from different layers were then clustered again in the second step of the algorithm for computing the individual tree locations. Monnier et al. [21] focused on detecting individual trees from a point cloud that was collected with a mobile laser scanner for dense urban areas. In their study, trees were segmented using local geometrical features of individual points. Trunks of trees were assumed to exist in the point cloud and were approximated by vertical cylinders to generate a "cylindrical descriptor." Individual trees were detected by combining the information from both the cylindrical descriptors and the segmented tree points.

However, these methods suffer from various problems when used to identify individual tree locations and extract related features from the photogrammetric-generated point cloud. For example, trunks of trees may not exist due to the dense canopy and leaves inside of crowns cannot be collected since photogrammetric techniques do not have the penetration capability of LIDAR does. In addition, treetop surfaces may not always form a regular shape such as a parabolic surface due to data noise and the lack of 3D reconstruction accuracy.

3. Research objective and questions

The objective of this research is to lay the groundwork for the semantic labeling of 3D point clouds/meshes in integrating the proposed framework into the existing workflow for the creation of virtual environments and simulations. As such, specific research questions that need to be answered include the following.

1. How should photogrammetric-generated point clouds be classified into top-level terrain elements (i.e., ground, buildings, and vegetation) considering the data quality issues and lack of point features compared with LIDAR-generated point clouds?
2. How should individual tree locations be identified using the classified point cloud considering the fact that data of tree trunks may be missing?

4. Research methodology

The proposed framework is presented in Fig. 1, which emphasizes the main elements and steps involved in the process. It is designed based on the review of the literature as stated in Section 2 in which top-level terrain elements (i.e., ground, trees, and buildings) are segmented before individual tree locations identification process can take place. Since

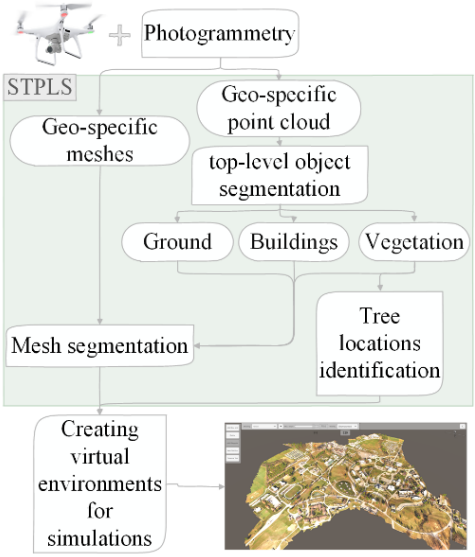


***FIGURE* 1: Semantic Terrain Points Labeling Framework**

the input of a virtual environment and simulation need to be in a mesh format instead of a point cloud format, mesh segmentation is a necessary step. Details of each step in the framework are discussed in the following sections.

4.1. Top-level objects segmentation workflow design

In order to understand and evaluate the effectiveness and performance of different classifiers in the context of top-level terrain elements segmentation, a designed point-cloud segmentation workflow is shown in Fig. 2. This workflow is designed following previous studies that focused on LIDAR point cloud segmentation [16]. The workflow utilizes both supervised and unsupervised machine learning processes. Since a very dense point cloud is usually generated with the photogrammetric technique, a down sampling process needs to be performed to accelerate further processes. It is worth noting here that the various point densities that are presented in the raw photogrammetric-generated point cloud is due to the way of data collection, not because of different objects presented in the point cloud. For instance, areas with dense points in a raw point cloud may be caused by more images captured within that area. Furthermore, non-uniformly distributed point spacing will affect the performance of extracting local point features in subsequent processes. A voxelization algorithm is used for down sampling the raw point cloud and to ensure that point spacing is uniformly distributed. To voxelize a point cloud, the 3D space is

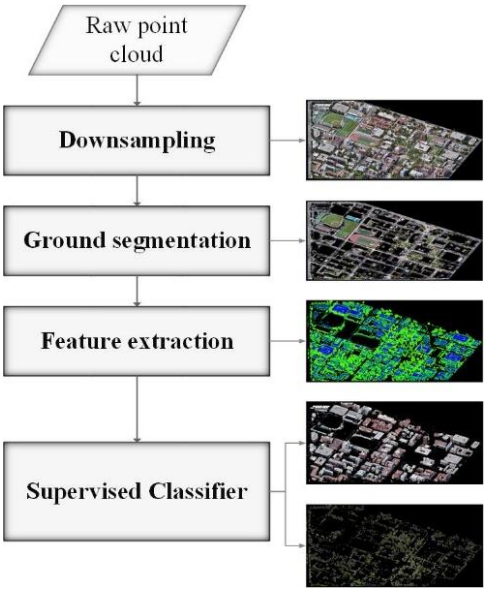


***FIGURE* 2. Workflow of Top-level Point Cloud Segmentation**

first discretized into 3D grids with defined grid spacing (e.g., 0.3 meters). Points that then fall into one grid are replaced with the centroid of that grid.

The second step in the designed workflow is grounds extraction. The reason that ground points should not be classified with supervised learning algorithms in this case is that point clouds generated with photogrammetric technique do not have the echo-based and full-waveform-based features like LIDAR-generated point clouds do. As such, it is a challenging task to classify ground points with only local point descriptors. Local point descriptors will be discussed in the following section. For instance, roof points have very similar descriptors to ground points such as planarity and point density ratio. Sithole and Vosselman [25] compared eight different ground segmentation filters and concluded that all had better performance in smooth rural landscapes than in complex urban areas and rough terrain with vegetation. In this study, the designed ground extraction process combined the use of region growing and a progressive morphological filtering algorithm to overcome the limitations of each individual approach for complex urban areas. The region-growing process is similar to flood fill for 2D image processing. It starts with a random seed point and examines all neighboring points within a defined radius. Points with similar normal values are added to the current cluster. The process runs recursively until no more points can be added. The entire region- growing process is terminated when all points are inspected. The largest cluster is then identified as ground. Two limitations of using the region-growing algorithm to extract ground points include: (1) the algorithm will fail to extract ground isolated by walls or buildings (e.g., courtyards); and (2) the algorithm will not work with a sloped terrain point cloud. To overcome such limitations, a progressive morphological filter algorithm was adopted to extract the isolated and sloped ground points. The progressive morphological filter algorithm was originally proposed by Zhang, et al. for segmenting airborne LIDAR data [17]. It is designed based on mathematical morphology. The operations in mathematical morphology include “dilation” and “erosion,” “opening,” and “closing” [26]. The core concept of mathematical morphology is to retain points that cannot be affected by a combination of abovementioned operations. One limitation of mathematical morphology is that a fixed window size has to be predefined, which means that noises with a larger size than the defined window cannot be removed. With the progressive morphological filter algorithm, the window size does not need to be predefined and will be iteratively

increased during the operations to remove all noise (i.e., vegetation and building points in a point cloud).

The next two steps in the designed workflow follow a common supervised machine-learning process. Since the segmentation needs to be performed on a point-level, point descriptors need to be computed for each individual point. Following that, with the defined point descriptors, the supervised classifier will be used to classify the points into different categories. A classifier needs a training using a set of manually classified points; the trained classifier can then be used to classify the unlabeled points.

4.1.1. Point descriptors

The use of effective point descriptors is the foundation of obtaining an accurate classification result. Previous studies have demonstrated the effectiveness of using local point descriptors for segmenting a LIDAR-generated point cloud [27]. Local point descriptors are computed for each point. Such a descriptor of a single point can be derived by analyzing its surrounding points within a predefined radius. For instance, geometric features such as planarity and curvature can be computed using eigenvalues derived from a principal component analysis of the surrounding points. As one can imagine, these descriptors can be very helpful for identifying whether a point falls on a flat surface or edge. For a more precise point description, point feature descriptors in each category are computed in a multi-scale fashion by varying the radius for selecting surrounding points. Analyzing points in multi-scales have been proven to be more robust to the presence of noise and offer more detailed local surface information [28]. For instance, if considering a point that falls on the edge of a window frame, the planarity would be low when considering the surrounding points within 10 centimeters, but high when considering the surrounding points within 1 meter. Details of the local point descriptors are discussed in the following.

***Color-based descriptors*:** Each point in a photogrammetric-generated point cloud contains color values represented as red, green, and blue (RGB). The color values are transformed from RGB color space to (hue, saturation, value) HSV color space since it has been proved that HSV color space is more suitable for color image segmentation [29]. Three-point features—i.e., the average, standard deviation and variance—are computed at each scale for every color channel.

***Point density-based descriptors*:** The downsampled point cloud has a uniformly distributed point density as previously discussed. Three object-related point density features are computed at each scale including: (1) the number of points n in a sphere with a predefined radius *r*; (2) the number of points m in a cylinder with the same radius *r* and a fixed height *h*; and (3) the point density ratio computed by taking the ratio between *n* and *m*.

***Local surface-based descriptors*:** The local surface-based features of each point data are computed using the eigenvalues that are derived from the covariance matrix of its *n* local surrounding points in the sphere. The covariance matrix is calculated with

$$\sum \mathrm{p} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} (\mathrm{p}^{\mathrm{i}} - \bar{\mathrm{p}})(\mathrm{p}^{\mathrm{i}} - \bar{\mathrm{p}})^{\mathrm{T}}, \qquad (1)$$

where p is the point data that is represented using its x, y, and z coordinates; p^{i} is one of its n surrounding points; $\bar{\mathrm{p}}$ is the mean/center of its surrounding points. The eigenvalues $\lambda 1 > \lambda 2 > \lambda 3$ are then computed with a principal component analysis based on the covariance matrix. Please note that eigenvalues need to be normalized between 0 to 1 with respect to $\lambda 1$. The local surface-based features include the following:

$$\text{Omnivariance} = \sqrt[3]{\prod_{\mathrm{i}=1}^{3} \lambda_{\mathrm{i}}} \qquad (2)$$

$$\text{Eigen entropy} = -\sum_{\mathrm{i}=1}^{3} \lambda_{\mathrm{i}} \ln(\lambda_{\mathrm{i}}) \qquad (3)$$

$$\text{Anisotropy} = \frac{(\lambda_1 - \lambda_3)}{\lambda_1} \qquad (4)$$

$$\text{Planarity} = \frac{(\lambda_2 - \lambda_3)}{\lambda_1} \qquad (5)$$

$$\text{Sphericity} = \frac{(\lambda_3)}{\lambda_1} \qquad (6)$$

$$\text{Linearity} = \frac{(\lambda_1 - \lambda_2)}{\lambda_1} \qquad (7)$$

$$\text{Curvature} = \frac{\lambda_3}{(\lambda_1 + \lambda_2 + \lambda_3)} \qquad (8)$$

$$\text{Verticality} = 1 - |\langle [0,0,1], e_3 \rangle| . \qquad (9)$$

Eigenvalues represent the magnitude in the direction where p 's neighboring points are extended. Different local surface point descriptors can be computed with the combination of the three eigenvalues as shown above. For instance, if a point lays on a planar surface, it is expected that its planarity to be close to 1 according to equation 5 since it's $\lambda 1$ and $\lambda 2$ will have similar magnitude but $\lambda 3$ will be much smaller than $\lambda 1$ and $\lambda 2$. These features can provide useful information for classifying buildings and trees. In the case of wall points, it is expected that they have large planarity and verticality values. Roof points have large planarity value but small verticality. Tree points have large eigen entropy value but small planarity value. It is worth noting here that thresholds are not used for classification process, instead, a supervised machine learning process is adopted.

4.1.2. Classifiers

Since photogrammetric-generated point clouds are different from LIDAR point clouds as previously discussed, the performance of different classifiers needs to be tested (i.e., classification accuracy and run-time efficiency). In this research, two supervised machine-learning algorithms used to solve classification problems with a similar nature were compared that include:

Support vector machine (**SVM):** C. Cortes and V. Vapnik explored statistical learning theory and developed the SVM algorithm with kernel methods and soft margin hyperplanes in 1995 [30]. Based on the study conducted by Weinmann et al., SVM achieved better accuracy than a naive Bayesian classifier in the case of a LIDAR point clouds urban scene classification [27]. As pointed out by Hsu, Chih-Wei et al., SVM-parameter tuning is an essential step [31]. The accuracy of the result highly depends on the selection of the parameters for the SVM. Thus, we followed the SVM-parameter-tuning guide that was provided by Hsu, Chih-Wei et al.

Random Forest: Quinlan, J. Ross originally proposed C4.5 to overcome some of the limitations presented in ID3, such as the inability to classify objects with continue attributes and address missing attributes [32]. Ho, Tin Kam introduced a random-forest method to construct a set of decision trees for a classification work to improve the accuracy and prevent overfitting [33]. Furthermore, the random-forest algorithm has the ability to rank the importance of each attribute and make selections [34]. Comparing to SVM, random forest is non-parametric and scale invariant, so feature normalization is not required.

4.2. Identify individual tree locations

One major limitation of using the photogrammetric technique to reconstruct 3D models for simulation is that it cannot create accurate vegetation models because of the extremely complex geometric properties of vegetation [3]. Such a limitation not only causes the vegetation visual appearance to be poor in a virtual environment, but it also limits the simulation functionalities such as computing the shortest path from a start point to a destination. For instance, when computing the shortest path going through a group of trees in a photogrammetric-generated virtual environment, the path cannot be accurately computed since the reconstructed tree models appear as a big solid blob instead of individual trees [3]. The path will be computed to either go over or around the trees and, as such, both cases are incorrect. Fig. 3 shows an example of such a scenario in the authors previously developed simulation tool —i.e., Aerial Terrain Lineofsight Analysis System (ATLAS), where blue icon is representing the start point of a soldier, pin icon is representing his/her destination, and the green line is representing the shortest path that was computed using the A* algorithm. This shortest path is not optimal due to the assumption that a unit cannot penetrate the 3D meshes and the fact that mesh tree canopies are directly connected to the ground mesh, the optimal shortest path is indicated as red line that goes through trees.

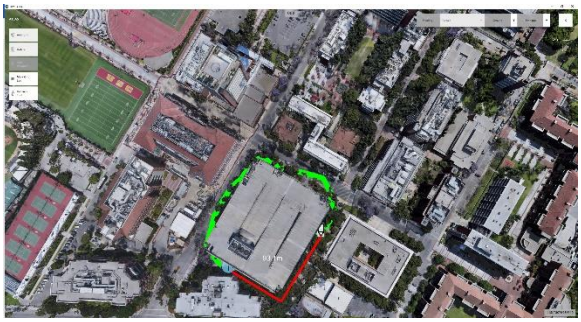


***FIGURE* 3. Incorrect Shortest Path**

One way of solving the abovementioned issue is to replace tree meshes with geo-typical 3D tree models. Such a process requires the extraction of information on individual tree locations and related features from the reconstructed 3D point cloud. Based on the literature review, the problem of identifying individual tree locations from LIDAR-generated point cloud can be considered as a model fitting problem [21], [23] or as a clustering problem [20], [22], [24]. However, as discussed in Section 2, a tree trunk may not exist, and treetop surfaces may not always form regular shapes, such as parabolic surfaces, due to the limitations of the photogrammetric technique. Thus, this research considers the problem of identifying individual tree locations in a photogrammetric-generated point cloud as a clustering issue. Identify individual tree locations by clustering points into different clusters should not rely on the assumption of tree trunk or treetop surface

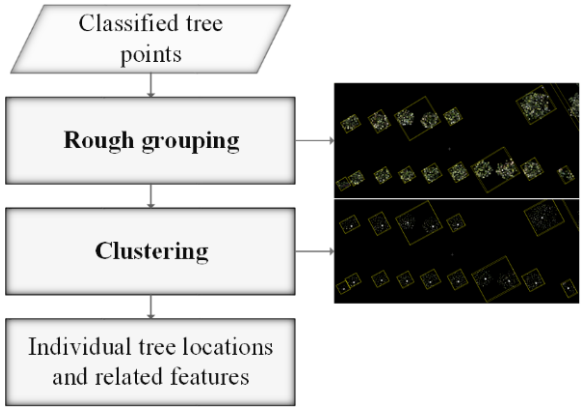


***FIGURE* 4. Workflow of Individual Tree Locations Identification**

shapes but utilizing the fact that points belong to one tree are close to each other.

The proposed clustering algorithm mainly consists of two steps as shown in Fig. 4. During the first step, the classified tree points are roughly segmented into different groups using a connected component-labeling algorithm. A defined Euclidean distance between points is set as a constraint for this rough segmentation process. As previously mentioned, the point cloud is down sampled with a user-defined point spacing that is data dependent. Adding a constant to the defined point spacing derives the distance used for the connected component segmentation process. There are generally two cases presented in the segmented groups: (1) only one tree is presented in a group; and (2) several trees are segmented into one group. The problem that needs to be solved in the second step is to cluster points that are in one group in case (2) into different clusters, and each cluster should be representing an individual tree. The k-means clustering algorithm was selected in this research, in which the center of each extracted cluster is considered as a tree location. One constraint embedded into the algorithm is that the center of each cluster has to be on one of the points in the tree point cloud. Furthermore, to segment points into different clusters using the k-means algorithm, finding the k value (i.e., number of clusters) is a critical step. The number of clusters in this case also represents the number of trees in a group. The proposed strategy is to run the k-means algorithm several times and increase the k value at each time until a pre-defined maximum point-to-center distance is satisfied for every point to the center of its belonging cluster. Intuitively, the maximum point-to-center distance is the average tree width. Related features such as color, the width, and height of a tree, can then be extracted from the points in each cluster.

4.3. Mesh segmentation and implementation

The proposed top-level object classification framework was implemented with C++ and python. The Point Cloud Library (PCL) was used for downsampling a raw point cloud and extracting point features. The SVM and random forest algorithms were implemented in Scikit-learn for the classification process. Note that both algorithms were implemented with parallel processing to accelerate the classification process. Individual tree location identification was implemented using python 2.7.

The photogrammetric technique can generate a mesh in the same coordinate system with the point cloud. Thus, meshes can be segmented by using a nearest neighbor algorithm to keep a set of vertices that are close enough to points in the segmented point cloud. Following that, edges in the mesh will be kept if both of its linking vertices are kept; mesh edges will be eliminated if one of its linking vertices is excluded. The closeness can be derived using the point density that was used for the point cloud down sampling process. This closeness will be further discussed in the case study.

5. Case study

A case study was conducted for the University Park Campus of the University of Southern California (USC). Two classifiers (i.e., the SVM and random forest) were compared based on the classification result. Following that, individual tree locations were extracted.

5.1. Data collection

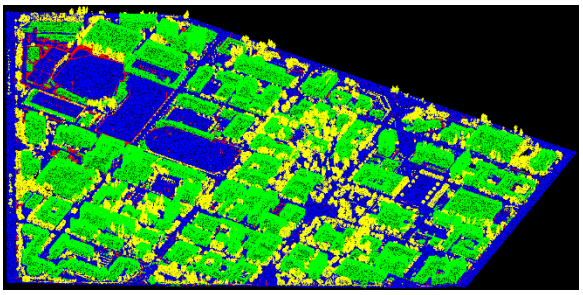
USC is located two miles south of downtown Los Angeles. This campus covers a total of 308 acres and consists of 159 buildings, trees and grassland, and paved ground (including vehicle roads, pedestrian roads, and squares). Buildings are typically an average height of 5–6 floors, with various appearances, colors, and shapes. Approximately 20% of the USC campus is covered by tree canopy.

Images were captured with a DJI Phantom 4 Pro. Note that, the flight planner—i.e., Rapid Aerial PhoTogrammetric Reconstruction System (RAPTRS)— was used for the flight path planning. The RAPTRS was designed under the OWT project for imaging large areas across multiple flights. Details of the RAPTRS can be found at [3]. Camera orientation and overlap between images were set to 45 degrees forward and 75%, respectively. Flight altitude was set to 65 meters. The point clouds were generated with ContextCapture (i.e., a photogrammetry software). The point clouds were downsampled to 3.8 million points (0.5-meter point spacing).

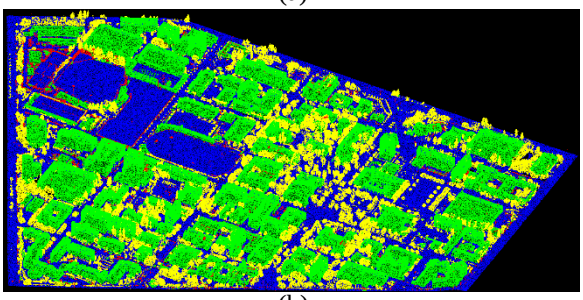
5.2. SVM vs. random forest for top-level object classification

To create the training data set, 20% of the points are manually classified. Classes that are classified include (1) ground; (2) buildings; (3) trees; and (4) others, which includes points that belong to light poles, fences, cars, and so forth. The classification results for the USC data set are shown in Fig. 5. The ground, buildings, trees, and others are marked with blue, green, yellow, and red, respectively.

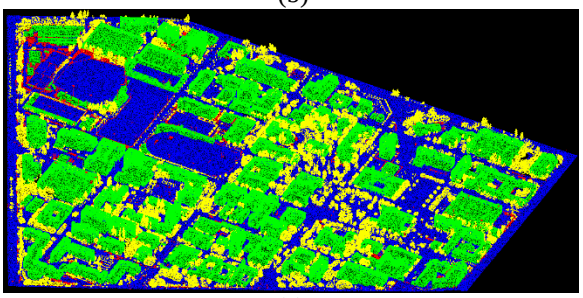
The confusion matrixes of the classification results are shown in Table 1 and 2. The SVM and random



(a)



(b)



(c)

***FIGURE* 5. Classification results of USC data set. (a) Ground truth; (b) Classified with SVM; and (c) Classified with random forest**

forest algorithms produced very similar results. The overall accuracy of SVM and random forest are 0.92 and 0.91, respectively. The SVM algorithm outperformed the random forest algorithm for classifying "buildings" and "others." Note that, the accuracy for classifying "others" with both the SVM and random forest algorithms are quite low (i.e., 0.57 when using SVM and 0.5 when using random forest).

***TABLE* 1. Confusion matrixes of RF classifier**

	Ground	Buildings	Trees	Others
Ground	0.92	0.03	0.03	0.02
Buildings	0.03	0.93	0.03	0.01
Trees	0.03	0.03	0.93	0.0
Others	0.15	0.25	0.1	0.5

***TABLE* 2. Confusion matrixes of SVM classifier**

	Ground	Buildings	Trees	Others
Ground	0.92	0.03	0.03	0.02
Buildings	0.03	0.95	0.02	0.0
Trees	0.03	0.04	0.93	0.0
Others	0.15	0.21	0.08	0.57

This is due to the following two reasons: (1) there is not enough point data in the training data set for others. Thus, the number of points is not balanced in the training set, which means "others" contains much fewer data points compared to other categories such as buildings; and (2) there are several different objects that are contained in "others" such as cars and light poles, which are not similar in shape, color, and texture. The computation time for training an SVM classifier and a random forest classifier are 446s and 238s respectively. The computation time for using the SVM classifier and the random forest classifier to classify unseen data are 1447 s and 186 s respectively.

5.3. Mesh segmentation and tree location identification

The proposed mesh segmentation process was validated using the USC dataset. USC meshes were segmented based on the point cloud segmentation result. The distance/closeness for selecting mesh vertices is set to 1 meter during the mesh segmentation process. Note that, this distance needs to be larger than the down sampling point spacing (i.e., 0.5 meter) since some of the misclassified points could be sparsely falling on an object (e.g., a few points on building roofs are misclassified as tree points). If down sampled point spacing is used for the distance/closeness, the segmented meshes will contain small holes due to these misclassified points. Furthermore, segmenting trees from ground can create holes on the ground meshes in some cases. For instance, some of the tree

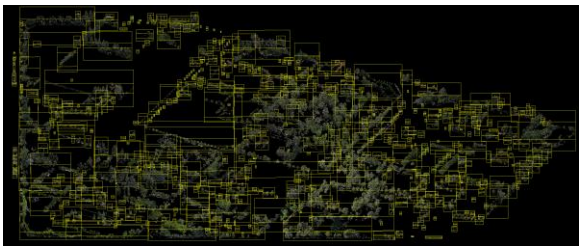
(a)

(b)

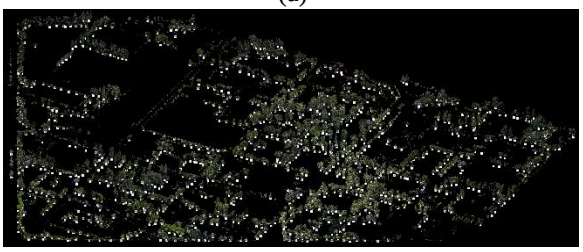
***FIGURE* 6. Mesh Segmentation (a) Segmented Buildings (b) Segmented Ground**

canopies are directly connected to the ground in the generated meshes due to the large size of a tree canopy and the limitation of photogrammetric technique. Thus, instead of segmenting tree meshes, they are flattened to the elevation of their closest ground mesh.

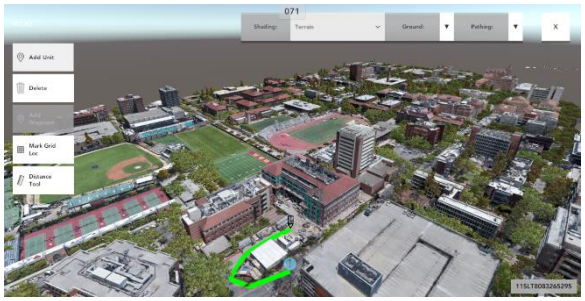
The proposed tree location identification process was performed on the USC dataset. The classification result from the SVM classifier is used for identifying individual tree locations and extracting building footprints. Each step of the proposed individual tree-location identification process is shown in Fig. 7. The clusters that were generated using the connected component algorithm are shown in Fig. 7 (a), where each yellow bounding box represents one cluster. The identified tree locations are shown in Fig. 7 (b), where each white point represents a tree location. Fig. 7 (c) shows the simulation environment that was generated using the segmented USC meshes. The mesh trees are replaced with geo-typical 3D tree models using the identified tree locations and related features. The average tree width was set to 7 meters for the k-means algorithm and the minimum number of points was set to 30 for the connected component algorithm.



(a)



(b)



(c)

***FIGURE* 7. Tree Locations Identification (a) Clustered Points (b) Individual tree locations (c) USC Dataset in Simulations Environment**

6. Discussion and Conclusion

As stated in the literature review, most previous studies on point cloud classification and information extraction have focused on LIDAR-collected data. In this study, a point cloud/mesh segmentation and information extraction framework for photogrammetric-generated point clouds/meshes is proposed. The proposed framework has been tested on USC data sets. The results indict that the SVM classifier outperforms the random forest classifier on classification accuracy. However, the running time for random forest is much shorter than for SVM. Thus, the SVM is recommended for an autonomous application where training data preexist and cannot be altered. The random forest classifier, on the other hand, is recommended for an interactive application where users can correct some of the miss-classified points and perform the classification process again to achieve better accuracy. The accuracy for classifying “others” in both cases are low, and further research on classifying small objects from large outdoor scenes is still needed. The results also showed that the proposed information extraction process could be integrated into the existing workflow of virtual environment and simulation creation. Three-dimensional tree models could also be placed at the identified tree locations to enhance the visual quality. However, as a quantitative analysis of the approach has yet not been accomplished, it will be a part of our future work.

10. References

[1] I. Brilakis *et al*, "Toward automated generation of parametric BIMs based on hybrid video and laser scanning data," *Advanced Engineering Informatics*, vol. 24, *(4)*, pp. 456-465, 2010.

[2] M. Golparvar-Fard, F. Peña-Mora and S. Savarese, "Sparse reconstruction and geo-registration of site photographs for as-built construction representation and automatic progress data collection," in *26th International Symposium on Automation and Robotics in Construction (ISARC 2009)*, 2009, .

[3] R. Spicer, R. McAlinden and D. Conover, "Producing usable simulation terrain data from UAS-collected imagery," in *Interservice/Industry Training, Simulation and Education Conference (I/ITSEC)*, 2016, .

[4] D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach.* 2011.

[5] M. Golparvar-Fard, F. Peña-Mora and S. Savarese, "D4AR–a 4-dimensional augmented reality model for automating construction progress monitoring data collection, processing and communication," *Journal of*

Information Technology in Construction, vol. 14, *(13)*, pp. 129-153, 2009.

[6] J. Armesto *et al*, "FEM modeling of structures based on close range digital photogrammetry," *Autom. Constr.*, vol. 18, *(5)*, pp. 559-569, 2009.

[7] I. Brilakis, H. Fathi and A. Rashidi, "Progressive 3D reconstruction of infrastructure with videogrammetry," *Autom. Constr.*, vol. 20, *(7)*, pp. 884-895, 2011.

[8] L. Klein, N. Li and B. Becerik-Gerber, "Imaged-based verification of as-built documentation of operational buildings," *Autom. Constr.*, vol. 21, pp. 161-171, 2012.

[9] A. Koutsoudis *et al*, "Multi-image 3D reconstruction data evaluation," *Journal of Cultural Heritage*, vol. 15, *(1)*, pp. 73-79, 2014.

[10] A. Rashidi, "Improved Monocular Videogrammetry for Generating 3D Dense Point Clouds of Built Infrastructure." , Georgia Institute of Technology, 2014.

[11] M. Himmelsbach, T. Luettel and H. Wuensche, "Real-time object classification in 3D point clouds using point feature histograms," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference On*, 2009, .

[12] L. Wallace *et al*, "Assessment of Forest Structure Using Two UAV Techniques: A Comparison of Airborne Laser Scanning and Structure from Motion (SfM) Point Clouds," *Forests*, vol. 7, *(3)*, pp. 62, 2016.

[13] A. Frome *et al*, "Recognizing objects in range data using regional point descriptors," *Computer Vision-ECCV 2004*, pp. 224-237, 2004.

[14] N. Chehata, L. Guo and C. Mallet, "Airborne lidar feature selection for urban classification using random forests," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 38, *(Part 3)*, pp. W8, 2009.

[15] K. Zhang *et al*, "A progressive morphological filter for removing nonground measurements from airborne LIDAR data," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, *(4)*, pp. 872-882, 2003.

[16] A. Boulch, B. Le Saux and N. Audebert, "Unstructured Point Cloud Semantic Labeling Using Deep Segmentation Networks," 2017.

[17] J. Huang and S. You, "Point cloud labeling using 3d convolutional neural network," in *Pattern Recognition (ICPR), 2016 23rd International Conference On*, 2016, .

[18] C. Zhang, Y. Zhou and F. Qiu, "Individual tree segmentation from LiDAR point clouds for urban forest inventory," *Remote Sensing*, vol. 7, *(6)*, pp. 7892-7913, 2015.

[19] F. Monnier, B. Vallet and B. Soheilian, "Trees detection from laser point clouds acquired in dense urban areas by a mobile mapping system," *Proceedings of the ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS Annals), Melbourne, Australia*, vol. 25, pp. 245-250, 2012.

[20] Y. Huang *et al*, "Toward automatic estimation of urban green volume using airborne LiDAR data and high resolution Remote Sensing images," *Frontiers of Earth Science*, vol. 7, *(1)*, pp. 43-54, 2013.

[21] A. Persson, J. Holmgren and U. Soderman, "Detecting and measuring individual trees using an airborne laser scanner," *Photogramm. Eng. Remote Sensing*, vol. 68, *(9)*, pp. 925-932, 2002.

[22] T. Ritter *et al*, "Automatic Mapping of Forest Stands Based on Three-Dimensional Point Clouds Derived from Terrestrial Laser-Scanning," *Forests*, vol. 8, *(8)*, pp. 265, 2017.

[23] G. Sithole and G. Vosselman, "Experimental comparison of filter algorithms for bare-Earth extraction from airborne laser scanning point clouds," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 59, *(1)*, pp. 85-101, 2004.

[24] R. Honsberger, *Mathematical Gems II Dolcani Mathematical Expositions 2*. 1976.

[25] M. Weinmann, B. Jutzi and C. Mallet, "Feature relevance assessment for the semantic interpretation of 3D point cloud data," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 5, pp. W2, 2013.

[26] M. Pauly, R. Keiser and M. Gross, "Multi-scale feature extraction on Point-Sampled surfaces," in *Computer Graphics Forum*, 2003, .

[27] D. J. Bora, A. K. Gupta and F. A. Khan, "Comparing the performance of L* A* B* and HSV color spaces with respect to color image segmentation," *arXiv Preprint arXiv:1506.01472*, 2015.

[28] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learning*, vol. 20, *(3)*, pp. 273-297, 1995.

[29] C. Hsu, C. Chang and C. Lin, "A practical guide to support vector classification," 2003.

[30] J. R. Quinlan, *C4. 5: Programs for Machine Learning*. 1993*1*.

[31] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, *(8)*, pp. 832-844, 1998.

[32] T. K. Ho, "Random decision forests," in *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference On*, 1995, .